VITA

Q1 using System;

```
class Program
    {
        static void Main(string[] args)
        {
mycall("vita");
mycall("vita",55);
Console.ReadLine();
        }
        static void mycall(string message, int age =25)
        {
       Console.WriteLine("{0}", message);
            Console.WriteLine("{0}", age);
        }
    }
```

A. Vita,25 ,vita,55 ==answer
B. Vita,vita,55
C. Error
D. Vita,55,vita,25

Q2 using System;

```
  class Program
  {
     static void Main(string[] args)
     {
        DisplayFancyMessage(message: "vita",  age: 25,addr: "juhu");

       Console.ReadLine();
     }
    static void DisplayFancyMessage(int age,string message, string addr)
     {

     Console.WriteLine(message);
     Console.WriteLine("{0} {1}",age,addr);

     }

  }
}
```

A vita,25,juhu ===answer
b.error
c.juhu,vita,25
d runtime error

Q3 using System;

```
class Program
  {
     static void Main(string[] args)
     {
        DisplayFancyMessage(message= "vita",  age= 25,addr= "juhu");
       Console.ReadLine();
     }

    static void DisplayFancyMessage(int age,string message, string addr)
```

```
      {

      Console.WriteLine(message);
      Console.WriteLine("{0} {1}",age,addr);

      }
   }
}
```

.A.vita,juhu,25
b. Error ==answer
c.juhu,vita,25
d.runtime error

Q4 IClonable interface has abstract method
A Clone ===clone
b.memberwiseclone
c.both
d.None

Q5

```
class Program
   {
      static void Main(string[] args)
      {
         DisplayFancyMessage( "Wow! Very Fancy indeed!", 50, name:"raj");
       DisplayFancyMessage( "geeta",  message: "hello",50);
         Console.ReadLine();
      }
      static void DisplayFancyMessage( string message,int number, string name,)
      {        Console.WriteLine("{0},{1},{2}",number,name,message );
      }

   }
```

A.Error ==answer
B.50,geeta,hello
c.hello,geeta,50
d none

Q6 foreach loop internally calling
A.Iclonable
b.IEnumerable ===answer
c.both
d. none

Q7

```
using System;
class Program
   {
      static void Main(string[] args)
      {
        EnterLogData(message:”Error”,string owner = "Programmer", DateTime timeStamp =
DateTime.Now)
```

```
        Console.ReadLine();
    }


static void EnterLogData(string message,string owner = "Programmer", DateTime
timeStamp = DateTime.Now)
{
Console.Beep();
Console.WriteLine("{0}", message);
Console.WriteLine("{0}", owner);
Console.WriteLine("{0}", timeStamp);
}
```

A.Error=== Answer
b. Error,Programmer,02/06/2015
c.none
d. Programmer, Error, 02/06/2015

Q8.IComparable has abstract method
A.compareTo ==answer
b.compare
c.comparer
d.all the above

Q9 IComparer has abstract method
A.Clone
b.compare ====answer
c.comparer
d.none

Q10 Which statement is true

A. when you implement interface and use abstract method you must use public access modifier.
B. when you implement interface and use abstract method you may use public access modifier

1.only A is true===answer
2. both are true
3.only b is true
4.none

Q11 Which statement is true
A.MemberwiseClone() method copy value type bit by bit and for reference type use shallow copy
b. MemberwiseClone() method copy value type and reference type as shallow copy
1.only b is true
2.only a is true ===answer
3. none
4. both

Q12 To short array you have

a. static sort() method in Array class==answer
b. virtual sort() method in Array class
c. user have to write own algorithm
d. none

Q13 What will be the output

```
using System;
delegate int addition();
   class myclass
   {
      int a, b;
      public int add()
      {          return a + b;
      }
      public myclass(int a, int b) { a = a;b = b; }
   }
   class Program
   {
      static void Main(string[] args)
      {
         myclass m = new myclass(6,6);

         addition a=m.add;
      int r = a();
         Console.WriteLine(r);
         Console.ReadLine();
      }
   }
}
```

A. 0 ===answer
B. 12
C. Error
D. None

Q 14

```
using System;
   delegate int addition();
      class myclass
      {
         int a, b;
         public int add()
         {          return a + b;
         }
         public myclass(int a, int b) {this. a = a;this.b = b; }
      }
      class Program
      {
         static void Main(string[] args)
         {
            myclass m = new myclass(6,6);

            addition a=m.add;
         int r = a();
            Console.WriteLine(r);
            Console.ReadLine();
         }
      }
```

```
    }
```

A. 12 ==answer
B. None
C. Erroe
D. 0

Q14.

```
delegate int addition(int x,int y);
  class myclass
  {  public int add(int p,int q)
     {          return p + q;
     }
     public int mul(int p,int q)
     {
        return p * q;
     }

  }
  class Program
  {
     static void Main(string[] args)
     {
        myclass m = new myclass();

        addition a=m.add;
        addition b = m.mul;
        addition tot = a + b;

           int r = tot(3,5);
        Console.WriteLine(r);
        Console.ReadLine();
     }
  }
}
```

A.15 ==answer
bError
c.8,15
d.none

Q15
deligate is derived from
a.System.Deligate
b.System. MulticastDelegate==answer
C none
D from both

Q16.

```
int invocationCount = d1.GetInvocationList().GetLength(0);
the above code assume d1 is variable of a type deligate
```

A. This method give length of method bind with deligate

B. This method give list of method
C. None
D. This method give list of parameter of method==answer

Q17.readonly key are internally static
a. True
b. False ==answer
c. none

Q18 readonly key can not be used in method
a. true
b. False ==answer
c. none

Q19 Which statement is true

A.as operator is like a cast,if conversion not possible it will return null instead of raising exception
B as operator is like a cast,if conversion not possible it will raise exception

1.only A ===Answer
2. only B
3. both true
4. both false

Q20 Array.Sort() method use
a Quicksort algorithm.
B Heapsort algorithm
c.insertion sort algorithm.
d. all three depend on size of data ===answer

Q21 as operator perform only
a. reference conversion
b. nullable conversion
c. boxing conversion
d. all the bove ===answer